JC05 Rec'd PCT/PTO 25 MAR 2002

FORM PTO-1390 (REV. 12-2001) U S DEPARTMENT OF COMMERCE PATENT AND TRADEMARK OFFICE

TRANSMITTAL LETTER TO THE UNITED STATES DESIGNATED/ELECTED OFFICE (DO/EO/US) CONCERNING A FILING UNDER 35 U.S.C. 371

ATTORNEY'S DOCKET NUMBER 1406/52

U S APPLICATION NO. (If known, see 37 CFR 1 5 10/088988

INTERNATIONAL APPLICATION NO.	INTERNATIONAL FILING DATE	PRIORITY DATE CLAIMED
PCT/EP00/09267	21 September 2000 (21.09.00)	24 September 1999 (24.09.99)

TITLE OF INVENTION
METHOD AND DEVICE FOR PROCESSING CONDITIONAL JUMP INSTRUCTIONS IN A PROCESSOR WITH PIPELINED ARCHITECTURE

APPLICANT(S) FOR DO/EO/US INFINEON TECHNOLOGIES, AG and NIE, Xiaoning

Applicant herewith submits to the United States Designated/Elected Office (DO/EO/US) the following items and other information:

1. [X] This is a **FIRST** submission of items concerning a filing under 35 U.S.C. 371.
2. [ ] This is a **SECOND** or **SUBSEQUENT** submission of items concerning a filing under 35 U.S.C. 371.
3. [ ] This is an express request to begin national examination procedures (35 U.S.C. 371(f)). The submission must include items (5), (6), (9) and (21) indicated below.
4. [X] The US has been elected by the expiration of 19 months from the priority date (Article 31).
5. [X] A copy of the International Application as filed (35 U.S.C. 371(c)(2))
   a. [ ] is attached hereto (required only if not communicated by the International Bureau).
   b. [X] has been communicated by the International Bureau.
   c. [ ] is not required, as the application was filed in the United States Receiving Office (RO/US).
6. [X] An English language translation of the International Application as filed (35 U.S.C. 371(c)(2)).
   a. [X] is attached hereto.
   b. [ ] has been previously submitted under 35 U.S.C. 154(d)(4).
7. [X] Amendments to the claims of the International Aplication under PCT Article 19 (35 U.S.C. 371(c)(3))
   a. [ ] are attached hereto (required only if not communicated by the International Bureau).
   b. [ ] have been communicated by the International Bureau.
   c. [ ] have not been made; however, the time limit for making such amendments has NOT expired.
   d. [X] have not been made and will not be made.
8. [ ] An English language translation of the amendments to the claims under PCT Article 19 (35 U.S.C. 371 (c)(3)).
9. [X] An oath or declaration of the inventor(s) (35 U.S.C. 371(c)(4)).
10. [X] An English lanugage translation of the annexes of the International Preliminary Examination Report under PCT Article 36 (35 U.S.C. 371(c)(5)).

Items 11 to 20 below concern document(s) or information included:

11. [ ] An Information Disclosure Statement under 37 CFR 1.97 and 1.98.
12. [ ] An assignment document for recording. A separate cover sheet in compliance with 37 CFR 3.28 and 3.31 is included.
13. [X] A FIRST preliminary amendment.
14. [ ] A SECOND or SUBSEQUENT preliminary amend
15. [ ] A substitute specification.
16. [ ] A change of power of attorney and/or address letter.
17. [ ] A computer-readable form of the sequence listing in
18. [ ] A second copy of the published international application under 35 U.S.C. 154(d)(4).
19. [ ] A second copy of the English language translation of the international application under 35 U.S.C. 154(d)(4).
20. [X] Other items or information:

"Express Mail" mailing number ET871448669US
Date of Deposit 25 March 2002
I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail to Addressee" service under 37 C.F.R. 1.10 on the date indicated above and is addressed to the Commissioner for Patents, Washington, D.C. 20231
Paige E. Snyder
Paige E. Snyder

Copy of cover page of PCT Publication; copy of International Preliminary Examination Report; copy of International Search Report

10/088988 JC13 Rec'd PCT/PTO 25 MAR 2002

U.S. APPLICATION NO. (if known, see 37 CFR 1.5)	INTERNATIONAL APPLICATION NO	ATTORNEY'S DOCKET NUMBER
10/088988	PCT/EP00/09267	1406/52

21. [X] The following fees are submitted:	CALCULATIONS	PTO USE ONLY
BASIC NATIONAL FEE (37 CFR 1.492 (a) (1) - (5)):		
Neither international preliminary examination fee (37 CFR 1.482) nor international search fee (37 CFR 1.445(a)(2)) paid to USPTO and International Search Report not prepared by the EPO or JPO **$1040.00**		
International preliminary examination fee (37 CFR 1.482) not paid to USPTO but International Search Report prepared by the EPO or JPO **$890.00**		
International preliminary examination fee (37 CFR 1.482) not paid to USPTO but international search fee (37 CFR 1.445(a)(2)) paid to USPTO **$740.00**		
International preliminary examination fee (37 CFR 1.482) paid to USPTO but all claims did not satisfy provisions of PCT Article 33(1)-(4) **$710.00**		
International preliminary examination fee (37 CFR 1.482) paid to USPTO and all claims satisfied provisions of PCT Article 33(1)-(4) **$100.00**		
ENTER APPROPRIATE BASIC FEE AMOUNT =	$ 890.00	
Surcharge of **$130.00** for furnishing the oath or declaration later than [ ] 20 [ ] 30 months from the earliest claimed priority date (37 CFR 1.492(e)).	$ 0.00	

CLAIMS	NUMBER FILED	NUMBER EXTRA	RATE	$	
Total claims	3 - 20 =	0	x **$18.00**	$ 0.00	
Independent claims	2 - 3 =	0	x **$84.00**	$ 0.00	
MULTIPLE DEPENDENT CLAIM(S) (if applicable)			+ **$280.00**	$ 0.00	
TOTAL OF ABOVE CALCULATIONS =				$ 890.00	
[ ] Applicant claims small entity status. See 37 CFR 1.27. The fees indicated above are reduced by 1/2. +				$ 0.00	
SUBTOTAL =				$ 890.00	
Processing fee of **$130.00** for furnishing the English translation later than [ ] 20 [ ] 30 months from the earliest claimed priority date (37 CFR 1.492(f)).				$ 0.00	
TOTAL NATIONAL FEE =				$ 890.00	
Fee for recording the enclosed assignment (37 CFR 1.21(h)). The assignment must be accompanied by an appropriate cover sheet (37 CFR 3.28, 3.31). **$40.00** per property +				$ 0.00	
TOTAL FEES ENCLOSED =				$ 890.00	
				Amount to be refunded:	$
				charged:	$

a. [X] A check in the amount of $ 890.00 to cover the above fees is enclosed.

b. [ ] Please charge my Deposit Account No. ______________ in the amount of $ ___________ to cover the above fees. A duplicate copy of this sheet is enclosed.

c. [X] The Commissioner is hereby authorized to charge any additional fees which may be required, or credit any overpayment to Deposit Account No. 50-0426 A duplicate copy of this sheet is enclosed.

d. [ ] Fees are to be charged to a credit card. **WARNING:** Information on this form may become public. **Credit card information should not be included on this form.** Provide credit card information and authorization on PTO-2038.

NOTE: Where an appropriate time limit under 37 CFR 1.494 or 1.495 has not been met, a petition to revive (37 CFR 1.137 (a) or (b)) must be filed and granted to restore the application to pending status.

SEND ALL CORRESPONDENCE TO
Richard E. Jenkins
JENKINS & WILSON, P.A.
Suite 1400 University Tower
3100 Tower Boulevard
Durham, NC 27707
US

25297
PATENT TRADEMARK OFFICE

Richard E Jenkins
SIGNATURE

Richard E. Jenkins
NAME

28,428
REGISTRATION NUMBER

"Express Mail" mailing number ET871448669US
Date of Deposit 25 March 2002
I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail to Addressee" service under 37 C F R 1 10 on the date indicated above and is addressed to the Commissioner for Patents, Washington, D C 20231
Paige E Snyder
Paige E. Snyder

10088958 .072102

JC13 Rec'd PCT/PTO 25 MAR 2002

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: Xiaoning Nie

Group Art Unit: Not Assigned

Serial No.: Not Assigned

Examiner: Not Assigned

Filed: Herewith

Docket No.: 1406/52

For. METHOD AND DEVICE FOR PROCESSING CONDITIONAL JUMP INSTRUCTIONS IN A PROCESSOR WITH PIPELINED ARCHITECTURE

* * * * * * * * * * * * * * * * * * * *

PRELIMINARY AMENDMENT

Honorable Commissioner for Patents
BOX PCT
Washington, D.C. 20231

Dear Sir:

Kindly amend the subject application as follows:

IN THE SPECIFICATION:

Please insert the paragraph heading on page 1 of the English translation of the subject application, before line 7, as follows:

--Technical Field --.

Please insert the paragraph heading on page 1 of the English translation of the subject application, line 10, as follows:

--Background Art --.

Please insert the paragraph heading on page 2 of the English translation of the Annex to Form PCT/IPEA/409, before line 30, as follows:

--Summary of the Invention--.

Please insert the paragraph heading on page 4 of the English translation of the subject application, before line 1, as follows:

--Brief Description of the Drawings--.

Please insert the paragraph heading on page 4 of the English translation of the subject application, line 30, as follows:

--Detailed Description of the Invention--.

REMARKS

The amendments to the specification as set forth above are intended to clarify and set apart the various sections of the subject application.

Attached hereto is a marked-up version of the specification, which illustrates all of the changes made to the specification pursuant to 37 CFR §1 121. The attached page is captioned "**Version With Markings To Show Changes Made**". Deleted language is bracketed and added language is underlined.

The Commissioner is hereby authorized to charge any deficiencies or credit any overpayments in connection with the filing of this correspondence to Deposit Account No. **50-0426**.

Respectfully submitted,

JENKINS & WILSON, P.A.

Date: 3-25-02

By: Richard E Jenkins

Richard E. Jenkins
Reg. No.: 28,428

Suite 1400 University Tower
3100 Tower Boulevard
Durham, North Carolina 27707
Telephone: (919) 493-8000
Facsimile: (919) 419-0383

25297
PATENT TRADEMARK OFFICE

1406/52 REJ/lsg

Serial No.: Not yet assigned

Version With Markings To Show Changes Made

IN THE SPECIFICATION:

The paragraph heading has been inserted on page 1 of the English translation of the subject application, before line 7, as follows:

Technical Field

The paragraph heading has been inserted on page 1 of the English translation of the subject application, line 10, as follows:

Background Art

The paragraph heading has been inserted on page 2 of the English translation of the Annex to Form PCT/IPEA/409, before line 30, as follows:

Summary of the Invention

The paragraph heading has been inserted on page 4 of the English translation of the subject application, before line 1, as follows:

Brief Description of the Drawings

The paragraph heading has been inserted on page 4 of the English translation of the subject application, line 30, as follows:

Detailed Description of the Invention

10/088988
JC13 Rec'd PCT/PTO 25 MAR 2002

Patent claims

1. Method for processing conditional jump instructions in a processor with pipeline computer architecture, that has the following steps:

a) loading and decoding a processor instruction, the processor instruction containing an instruction opcode, register addresses, a relative jump distance, a precondition and a post-condition,

b) execution of the decoded processor instruction if the precondition is fulfilled, and

c) jumping to a jump address as a function of the relative jump distance contained in the processor instruction if the post-condition is fulfilled.

2. The method as claimed in claim 1, in which the post-condition comprises a plurality of post-condition bits that are checked in the processor.

3. An apparatus for processing conditional jump instructions in a processor with pipeline computer architecture, having:

an instruction decoder (20) for decoding a processor instruction that contains an instruction opcode, register addresses, a relative jump distance, a precondition and a post-condition,

the instruction decoder (20) checking in the case of a fulfilled precondition whether the post-condition is fulfilled and, in the case of a fulfilled post-condition, driving a program counter (14) for forming a jump address as a

function of the relative jump distance contained in the processor instruction.

Description

Method and apparatus for processing conditional jump instructions in a processor with pipelined architecture

The present invention relates to a method and an apparatus for processing conditional jump instructions in a processor with pipelined architecture.

The number of cycles required for executing specific instructions is one of the most important performance parameters of a processor. The number of cycles is to be minimized as far as possible in order to achieve maximum processing speed and minimum power consumption. Processors with what is termed pipelined architecture are already known for this purpose in the prior art. This means that the processor processes a plurality of instructions simultaneously, each instruction being in a different stage of processing. For example, one instruction is just being executed, the next is simultaneously already decoded, the next but one has been requested from the memory, etc.

It is possible, in particular, in such a pipelined architecture for a conditional jump instruction (branch) to lead to what is termed a harzard, as a result of which it is even possible for wrong results to be produced. Specifically, in the case of a conditional jump instruction, the address of the next instruction is not fixed until after processing of this conditional jump instruction. In this way, therefore, the next instruction can be requested from the memory and decoded only once the result of the execution of the preceding instruction is available from the arithmetic-logic unit of the processor.

In accordance with the prior art, this hazard problem has been solved in such a way that, directly after the

remain correct in any case. However, not as many processor cycles are thereby utilized as dummy instructions that need to be processed.

M.J. MAHON ET AL.: 'HEWLETT-PACKARD PRECISION ARCHITECTURE: THE PROCESSOR' HEWLETT PACKARD JOURNAL, HEWLETT-PACKARD CO. PALO ALTO, US, Vol. 37, No. 8, August 1, 1986 (1986-08-01), pages 4 - 22, XP000211314 disclose, inter alia, that in the case of the execution of a branching instruction or jump instruction in a processor with pipeline processing of the instructions, a delay instruction is inserted following the jump instruction in order to permit the calculation of a jump destination address before a destination instruction of the jump is loaded, or the program flow runs further to the destination instruction. The delay instruction is not executed if the same is canceled by nullification by the immediately preceding jump instruction. In the case of nullification, an instruction that immediately follows a jump instruction is executed as NOP. All jump instructions have for this purpose a 1-bit nullification field that controls the nullification and thus the activation or deactivation of the delay instruction as a function of a jump instruction, in order to optimize the use of the delay instruction in jump instructions.

It is therefore the object of the present invention to permit the processing of conditional jump instructions in a processor with pipelined architecture without so great a loss of processor cycles by dummy instructions.

This object is achieved by means of a method as claimed in claim 1 and an apparatus as claimed in claim 3.

According to the invention, this object is achieved by means of a method for processing conditional jump instructions in a processor with pipelined architecture in the case of which there are added to each instruction according to which a conditional jump is to be executed one or more additional bits that specify under which condition the conditional jump is to be executed. It is already possible in this way to establish earlier an instruction as to whether a branch is to be carried out or not. Consequently, an instruction which will be the next instruction after the conditional jump is already fixed earlier. It is therefore possible to establish the jump destination of a conditional jump instruction much earlier by means of this branch prediction in the instruction set.

It is particularly preferred in this case that in addition the appropriate jump address is added to each instruction according to which a conditional jump is to be executed. In this way, not only is an instruction known earlier as to whether a conditional jump is to be carried out or not, but the corresponding new destination address is already known. The correct instruction can therefore already be requested from the main memory of the processor.

Furthermore, it is preferably possible in addition to add to each instruction one or more bits that specify

under which conditions the instruction is actually to be executed.

AMENDED SHEET

The present invention is explained in more detail below with the aid of the drawings attached in the enclosure, in which:

Fig. 1 shows the operating cycle of a processor with a two-stage pipeline;

Fig. 2 shows the design of a 22-bit long machine instruction according to the invention;

Fig. 3 shows the design of a 25-bit long machine instruction according to the invention;

Fig. 4 shows a schematic of an apparatus according to the invention for altering the program counter reading for the purpose of executing conditional jumps;

Fig. 5 shows a further apparatus according to the invention for altering the program counter reading for the purpose of executing conditional jumps;

Fig. 6 shows a schematic of the overall design of a processor with pipelined architecture for the purpose of executing conditional jump instructions with the branch prediction according to the invention; and

Fig. 7 shows a detailed illustration of a processor with apparatuses for the branch prediction according to the invention.

The present invention proceeds from a pipelined architecture for a processor. This is described, for example, in the book entitled "Computer Organisation and Design" by Pattersen & Hennessy.

Put briefly, the pipelined architecture signifies the following:

Normally, each machine instruction is processed by a processor by means of the following operations:

1. Instruction fetch
2. Instruction decoding
3. Execution
4. Write back

It is already known in the prior art to have these operations run partially in parallel, for example by executing an instruction precisely when the next instruction is already being decoded. This mode of procedure is illustrated in figure 1 for a two-stage pipeline.

Thus, a processor uses the pipeline in order to process on average one instruction per processor cycle.

However, this pipelined architecture of the processor leads to problems whenever conditional jump instructions are to be executed. This problem is termed "branch harzard" in technical language. This means that a branch instruction, that is to say a conditional jump instruction, can show whether the next instruction is to be further processed or a jump is to be made to another destination address only after execution of the preceding instruction.

This problem is solved in the prior art by filling the clock pulse after the conditional jump instruction with a no-operation instruction, that is to say an instruction to wait for a processor cycle. Although it is certainly ensured in any event that the program continues to run correctly, one processor cycle is lost, and thus so is the maximum possible computer power. The prior art is to be explained in more detail with the aid of the following examples, which respectively treat the calculation of the absolute value of a number:

Firstly, there is the possibility of conditional execution, that is to say, for example:

```
/* A = |B|  */
LOAD  R1  B
COMPARE R1   0   /*if B ≥ 0, carry = 0 */
NEGATIVE R1  on-carry  /* negate if carry = 1 */
STORE  R1  A
```

This type of execution is possible, however, only if only a single instruction must be executed conditionally, and this instruction includes no jump. In the case of more complex functions or tasks that can no longer be represented only by one instruction, a conditional jump must respectively be performed, as illustrated in the following program. As may be seen from the boxed program section, a no-operation instruction must be inserted downstream of the two jump instructions (in the case of a two-stage pipeline, and with longer pipelines correspondingly more no-operation instructions:

```
LOAD        R1      B
COMPARE     R1      0
JUMP ON CARRY       L1
JUMP        L2
NO OP
L1:    NEGATIVE     R1
L2:    STORE        R1   A
```

Finally, there is also the option in the prior art of what is termed speculative execution. This means that only one option is executed in the hope of hitting the correct continuation with a probability of somewhat more than 50%. However, this requires a very considerable outlay on hardware, since it is then necessary, after all, for some instructions to be

"unraveled" should the supposition not apply. Moreover, processor cycles are lost nevertheless if there has been a "mis-estimation".

In accordance with the prior art, there has thus so far not been any suitable solution to this problem, that such a branch harzard, that is to say a problem with the conditional branching, has effected a loss in working cycles of the processor in pipelined architecture. According to the invention, this problem is now solved by means of a combination of an instruction with "conditional execution" and a "jump arithmetic" instruction, as follows:

The aim here, once again, is to consider a simple example, specifically the instruction "Add R2 to R1, if R1 then = 0, jump to L1". This problem is written as follows in "C":

```
R1 = R1 + R2
if (R1 = = 0)
     GO TO L1
L1: ......
```

According to the invention, the machine instruction ADD R1, R2, #JMP, ON ZERO is used for this purpose. #JMP means in this case the relative jump address to the entry point L1.

We thereby extend the instruction once by a post-condition to the known pre-condition. For example: P1, ADD R1, R2, #JMP, Q1.

Here, P1 means: execute R1=R1+R2 if P1 is fulfilled. According to the invention, Q1 means: execute jump by JMP if Q1 is fulfilled after the calculation of R1=R1+R2.

It would thereby be possible for the following "C" program:

```
if (A=1)
B = A;
else
C = A;
```

to be translated into machine code as follows:

```
LOAD              R1  A
Q1    TEST        R1  1 # L /* if A=1 jump L */
P1    Q1   STORE  R1  B
STORE             R1  C
```

Thus, according to the invention, it is possible to provide in the instruction coding both bits for preconditions and bits for post-conditions, as is illustrated by way of example in figs. 2 and 3.

Fig. 2 shows in this case a simplified example with an instruction of only 22 bits in length, one bit 1 being provided for the precondition, one bit 2 for the post-conditions, 8 bits 3 to 10 for the relative displacement and then, as usual, three bits each for the two register addresses and 6 bits for the instruction code.

It is usually necessary in reality to check a plurality of conditions as precondition and post-condition. Consequently, correspondingly more bits must be provided, as is illustrated in fig. 3.

In fig. 3, the bits 0 to 1 contain the information for post-condition, the bits 2 and 3 information for pre-conditions, and the bits 4 to 10 the relative jump address, that is to say the jump distance.

The method according to the invention can be used with particular effect in conjunction with a program loop, for example for the following "C" program:

```
for (i=1; i < 5; i++)  {
      x [i] = i;
}        /*  C program */
```

According to the invention, this can then be converted into the following substantially simplified machine program:

```
                Load      R1   5
                Load      R2   X   /*Address of X[5]*;

L1= STORE_INDEXED R2   R1   /* x[i] = i */
Q1  DECREMENT     R1   #1   L1
           ADD    R2   1       /*i = i+1*/
```

The post-condition Q1 means: conditional jump if the result R1=R1-1 is not 0.

A further example of the simplifications that can be achieved according to the invention when programming is the program, represented below, for processing a ring buffer.

In accordance with the prior art, this problem would have had to be programmed as follows:

```
TST (R3) #buffer_end    // ring buffer end reached
BNZ NEXT                // if no
NOP
LDI (R3) #buffer_start  // else set the pointer to
                           buffer again
```

According to the invention, the following two instructions suffice instead of this:

```
TST (R3) #buffer_end
LDI (R3) #buffer_start
```

However, it is to be borne in mind that this solution according to the invention cannot be applied to all loop structures. Loop structures of all sorts can, however, be programmed as follows according to the invention:

```
LDI (R4) #loop_cnt_minus_1 // init loop counter
WHILE_LOOP:
        FIRST_PC                  // code sequency
        SUBI (R4)  #1 #loop_flag  // decrement by 1 and
                                     indicate loop end
        BNZ WHILE_LOOP            // if not zero go to
                                     loop begin
```

According to the invention, instead of the usual subtraction machine instruction SUB, use is made of a machine instruction SUBI that is extended such that it has a flag bit that is used for the purpose of indicating a cycle before the conditional jump instruction BNZ, which is the correct branch in the case of the conditional jump, such that no loss of processor cycles occurs at all in the case of a two-stage pipeline. Instruction LDI indicates a loop start.

The typical solution for avoiding the branch harzard is based on predicting the expected jump destination of the conditional jump.

The implementation of a loop generally requires three steps:

1. Initialize the loop counter
2. Decrement or increment the loop counter
3. Jump to the end of the loop

The cycle loss in the case of the conditional jump is based on the fact that the next instruction that is executed after the jump depends on the fulfillment of the loop condition. Consequently, dummy instruction NOP must be inserted after the conditional jump instruction. By using a loop flag in an arithmetic instruction such as ADD or SUB, the loop condition can be checked at the end of the execution of the addition or subtraction instruction. The zero flag, that is to say the indication by the arithmetic-logic unit that it is at 0, can then be checked in order to decide to which address the program counter of the processor should be set. This LOOP flag can be interpreted as an ENABLE-DISABLE flag or, more generally, as an address displacement.

Figure 4 shows the simplest basic principle for implementing a LOOP flag in accordance with the invention.

The program memory 10 is connected in this case to the program counter 14 via a multiplexer 12. The output of the program counter (PC) 14 is connected to a logic gate 16 that combines the output value of the program counter with a constant or the LOOP flag. The output of this logic circuit 16 is connected to the input of the multiplexer (MUX) 12, whose other input is connected, after all, to the program memory 10, and whose output is connected to the program counter 14. The multiplexer 12 is controlled by a control signal (control) from the processor.

A further improvement to the invention permits dispensing with the jump instruction by buffering the start of the loop:

```
        LDP  (R4)  #loop_cnt_minus_1
    WHILE_LOOP:
         FIRST_PC
```

SUBI (R4) #1 #Loop-flag
NEXT_INS:

There is a need here for an additional instruction LDP indicating that a loop is starting. The next program code address is then buffered as loop start. The same result can also be achieved by using the instruction LDI and loading the next program counter value explicitly into the buffer. Of course, an additional instruction is, however, required again thereby. The instruction SUBI has a loop flag that is used to indicate which is the correct branch for the conditional jump. The zero flag is checked in order to decide whether the aim is to jump back to the start of the loop or to execute the next instruction (NEXT_INS) that is indicated by #-Loop-Flag.

This simplified processing of loop structures requires a somewhat more complicated structure of the circuit according to the invention, as illustrated in fig. 5.

As in fig. 4, here, as well, a program memory 10 is provided that is connected to the input of a multiplexer 12 whose output is connected, in turn, to the program counter (PC) 14. The output of the program counter (PC) 14 is likewise connected to a logic gate 16 that combines the output value of the program counter with the loop flag. The output of this logic circuit 16 is connected to a further input of the multiplexer (MUX) 12. However, in the present case, the multiplexer 12 has a further input, which is connected to a buffer 18 whose input can be loaded with the value of the program counter 14. The explicit instruction "Load the next program counter reading into the buffer" becomes superfluous in this way.

Fig. 6 shows the overall design of a processor with the ability to process the instructions according to the invention. Elements identical to those in figs. 4 and 5

are also provided with identical reference symbols. The program counter (PC) 14 once again accesses the program code memory 10, and respectively accesses the program line to be processed in this case. The corresponding instruction code is fed from the program memory 10 to the instruction decoder (IDEC) 20. The latter relays the corresponding control instructions to the arithmetic-logic unit (ALU) 22 and to the register set 24. The contents of the registers are then loaded if needed into the arithmetic-logic unit 22, or written back from there again, as indicated by the arrows. The flag signals zero, carry and overflow of the arithmetic-logic unit 22 are simultaneously fed both to the instruction decoder (IDEC) 20 and to the control input of the multiplexer (MUX) 12. The two inputs of the multiplexer 12 are occupied by the value 1 and by the relative jump value #JMP supplied by the instruction decoder 20. The output of the multiplexer 12 is connected to an adder 16, whose other input is connected to the output of the program counter 14.

It is to be borne in mind in the case of more than two pipeline stages that the flag signals zero, carry, overflow and the associated relative jump value #JMP must simultaneously be present at the multiplexer 12. However, this is not necessary in the case of a two-stage pipeline as described in the present exemplary embodiment. The corresponding instruction coding with the post-condition according to the invention is now described below. Reference may be made once again for this purpose to fig. 2, which illustrates the simplest possible instruction set according to the invention with a length of 22 bits.

The uppermost 6 bits (21 to 16) in this case contain the instruction code (OPCODE), for example: addition. The next three bits contain the address of the first register (REG A) with a length of three bits (conventional processors mostly use no more than 8

registers) on the bits 15, 14, 13, this being followed by the register address of the second, to be added in this case, register (REG B) on the bits 12, 11 and 10.

In the case of this instruction, the arithmetic-logic unit of the processor will therefore add the contents of the registers A and B and store them in the register. According to the invention, other bits are now added to this instruction, specifically the bits 9 to 2 (displacement), which specify the relative jump distance in the case of a following conditional jump. The condition bits 1 and 0 then follow, the bit 1 (post) indicating the post-condition, while the bit 0 (PRE) indicates the precondition.

The processing cycle is now as follows in this case: the instruction must be fetched and decoded. For this purpose, the processor starts at a specific program counter reading, for example PC=0.

Given this program counter reading, an instruction of 22 bits is fetched from the program memory, which is present at the address in the memory corresponding to this program counter reading.

The instruction is then processed by the instruction decoder (IDEC) 20.

An attempt is first to be made in this case to check whether the corresponding precondition bit is set. If this is the case, the instruction is not even executed given the non-fulfillment of the corresponding precondition.

The difference between the present invention and the prior art resides in the post-condition bits.

The signal "BR_CTR" is generated from this post-condition bits. The addition is carried out as follows, at the same time:

A control signal ALU-CTR and the read and write addresses and enable signals for the arithmetic-logic unit are generated. At the same time, the instruction decoder 20 makes the relative jump distance "BR" available. The "BR-CTR" signal drives the branch control according to the following prescriptions:

1. No jump if post-condition bit = 0, that is to say PCNEW=PCOLD+1

2. A relative jump is executed if the post-condition bit = 1 and the condition is fulfilled that, for example, zero flag = 1. The program counter 14 is therefore set to the new value PCNEW=PCOLD+BR.

If the post-condition bit is = 1, but the condition is not fulfilled, again no jump is carried out, that is to say: PCNEW=PCOLD+1.

It is possible to use more than one post-condition bit, as illustrated, for example, in fig. 3. It is then possible to check more conditions (for example zero, carry, overflow).

Thus, according to the invention, control information for the arithmetic-logic unit and information relating to jump destination addresses is simultaneously made available for the first time simultaneously by the instruction decoder 20 during decoding of the instructions.

The instruction is now executed and, if appropriate, branched.

The action of the arithmetic-logic unit (ALU) is executed for this purpose. The result is written back into the appropriate register. At the same time, the corresponding zero, carry, etc. flags are present at the output of the arithmetic-logic unit.

The bits for the individual flags, "BRCTR" and the value "BR" relating to the same clock pulse are made available in this case to the branching control. As illustrated in fig. 7, the control unit "Cond" 26 then generates two control signals S1 and S2. S1 prompts either to undertake no jump, or to work out a relative jump. S2 then switches through the relative jump address "PCNEW" through the multiplexer 12.

An additional instruction is thereby saved for the jump in addition to the corresponding arithmetic instruction. Consequently, a reduction can be achieved in the number of required instructions, and the throughput of the processor is raised thereby.

The design of a processor for processing instructions with the post-condition bits according to the invention is illustrated in detail in fig. 7. Identical numerals to those in figs. 4, 5 and 6 refer to identical units.

Also provided in fig. 7 is a program counter 14, which addresses an instruction memory (CODEROM) 10. From there, the instructions with an instruction width of 22 bits are fed to the instruction decoder (IDEC) 20. The latter generates the usual signals for driving the register 24 and the arithmetic-logic unit (ALU) 22. However, according to the invention it also generates in addition the signals "BR" (this signal comprises a plurality of bits) and specifies the relative jump distance as well as the signal "BR-CTR", which specifies that a conditional jump is to be processed and the corresponding flag bits of the arithmetic-logic unit are to be checked.

[redacted]

At its output, the arithmetic-logic unit 22 supplies results and the corresponding flags that illustrate specific conditions (for example, zero, overflow, carry, etc.). The results can, of course, also be fed to the registers 24 again. The "BR_CTR" signals and the flags from the ALU are fed to a further logic unit (Cond) 26. This generates as a function of the corresponding BR_CTR signals and the associated flags signals S1 and S2 that control the multiplexer 12 and a switch upstream of the one input of the adder 16. This switch switches over between 1 and "BR" as a function of the fulfillment of the flag conditions. The other input of this adder is connected to the output of the program counter 14.

It is possible in this way according to the invention to carry out a substantially faster processing of conditional jumps with a relatively low additional technical outlay at the processor.

(12) NACH DEM VERTRAG ÜBER DIE INTERNATIONALE ZUSAMMENARBEIT AUF DEM GEBIET DES PATENTWESENS (PCT) VERÖFFENTLICHTE INTERNATIONALE ANMELDUNG

(19) **Weltorganisation für geistiges Eigentum**
Internationales Büro

(43) **Internationales Veröffentlichungsdatum**
29. März 2001 (29.03.2001)

PCT

(10) **Internationale Veröffentlichungsnummer**
WO 01/22217 A1

(51) **Internationale Patentklassifikation**[7]: **G06F 9/32**, 9/38

(21) **Internationales Aktenzeichen:** PCT/EP00/09267

(22) **Internationales Anmeldedatum:** 21. September 2000 (21.09.2000)

(25) **Einreichungssprache:** Deutsch

(26) **Veröffentlichungssprache:** Deutsch

(30) **Angaben zur Priorität:**
199 45 940.1 24. September 1999 (24.09.1999) DE

(71) **Anmelder** *(für alle Bestimmungsstaaten mit Ausnahme von US)*: **INFINEON TECHNOLOGIES AG** [DE/DE]; St.-Martin-Strasse 53, 81669 München (DE).

(72) **Erfinder; und**
(75) **Erfinder/Anmelder** *(nur für US)*: **NIE, Xiaoning** [DE/DE]; Luitpoldring 41, 85591 Vaterstetten (DE).

(74) **Anwalt: BARTH, Stephan-Manuel**; Reinhard, Skuhra, Weise & Partner GbR, Postfach 440151, 80750 München (DE).

(81) **Bestimmungsstaaten** *(national)*: CN, JP, KR, US.

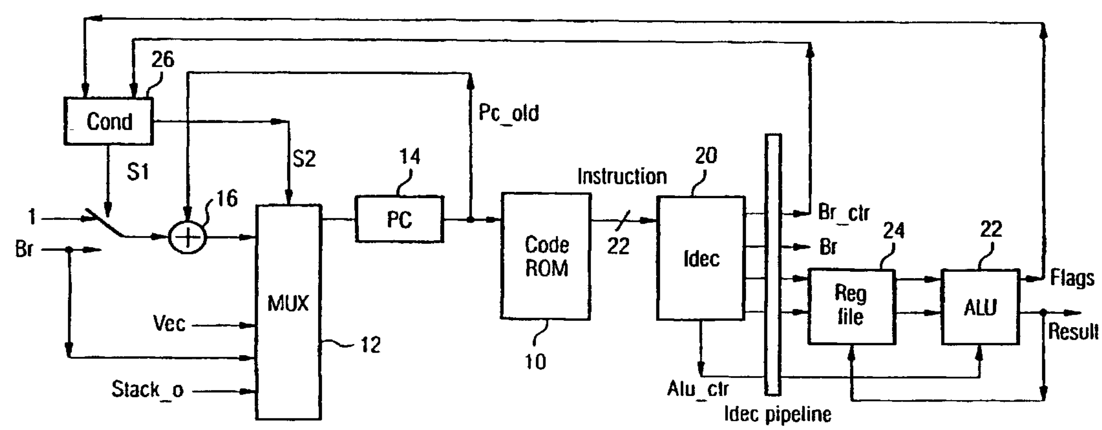
(84) **Bestimmungsstaaten** *(regional)*: europäisches Patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Veröffentlicht:
— *Mit internationalem Recherchenbericht.*
— *Vor Ablauf der für Änderungen der Ansprüche geltenden Frist; Veröffentlichung wird wiederholt, falls Änderungen eintreffen.*

Zur Erklärung der Zweibuchstaben-Codes, und der anderen Abkürzungen wird auf die Erklärungen ("Guidance Notes on Codes and Abbreviations") am Anfang jeder regulären Ausgabe der PCT-Gazette verwiesen.

(54) **Title:** METHOD AND DEVICE FOR PROCESSING CONDITIONAL JUMP INSTRUCTIONS IN A PROCESSOR WITH PIPELINED ARCHITECTURE

(54) **Bezeichnung:** VERFAHREN UND VORRICHTUNG ZUR BEARBEITUNG BEDINGTER SPRUNGBEFEHLE IN EINEM PROZESSOR MIT "PIPELINED"-ARCHITEKTUR



(57) **Abstract:** The invention relates to a method and a device for processing conditional jump instructions in a processor with pipelined architecture. One or more additional bits indicating the condition under which the conditional jump instruction is to be executed is/are added to each instruction stating that a conditional jump is to be executed. The inventive device can also comprise a device for altering the count of the program counter according to the additional bits for executing the conditional jumps.

(57) **Zusammenfassung:** Verfahren und Vorrichtung zur Bearbeitung bedingter Sprungbefehle in einem Prozessor mit "Pipelined"-Architektur, wobei jedem Befehl, nach dem ein bedingter Sprung ausgeführt werden soll, ein oder mehrere zusätzliche Bits hinzugefügt werden, die angeben, unter welcher Bedingung der bedingte Sprung auszuführen ist. Zusätzlich kann die Vorrichtung eine Vorrichtung zur Veränderung des Programmzählerstandes in Abhängigkeit von den zusätzlichen Bits zur Ausführung der bedingten Sprünge umfassen.

WO 01/22217 A1

FIG 1

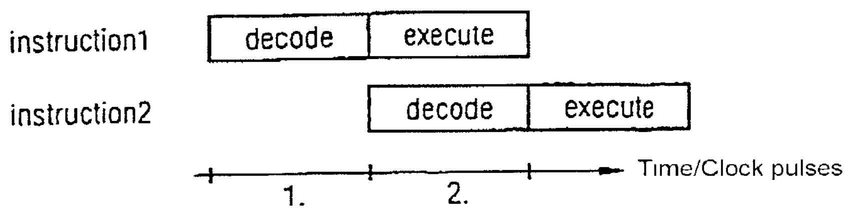


FIG 2

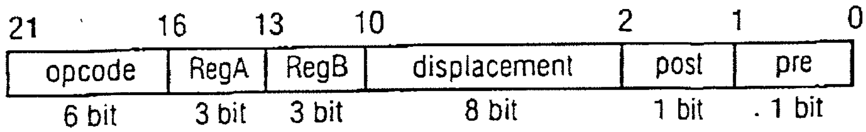


FIG 3

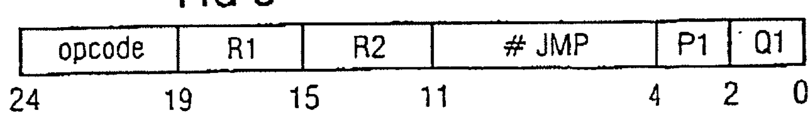


FIG 4

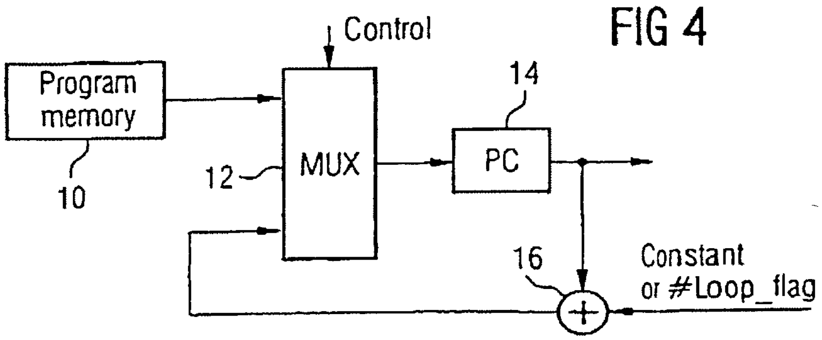


FIG 5

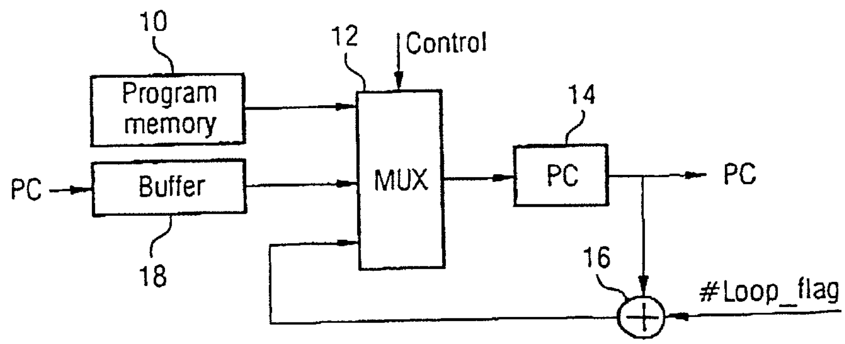

10
Program memory
12
Control
MUX
14
PC
PC
PC
Buffer
18
16
#Loop_flag

FIG 6

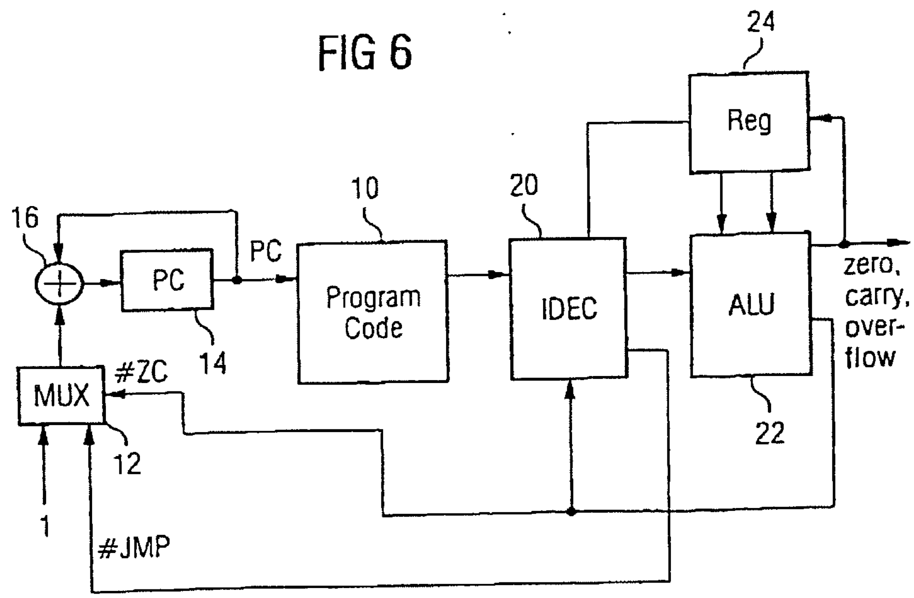
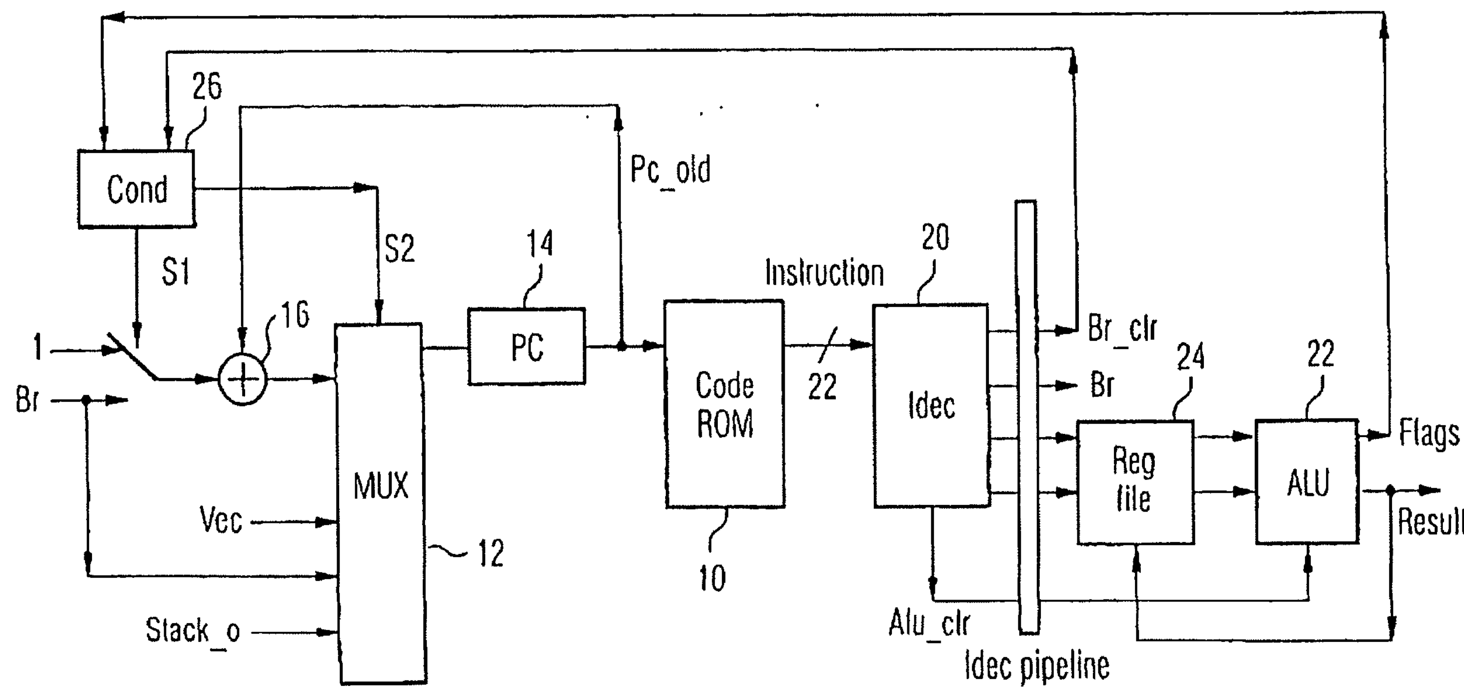

24
Reg
10
20
16
PC
PC
Program Code
IDEC
ALU
zero, carry, over- flow
14
#ZC
MUX
12
22
1
#JMP

FIG 7


Cond
26
S1
S2
1
Br
16
Vec
Stack_o
MUX
12
PC
14
Pc_old
Code
ROM
10
Instruction
22
Idec
20
Alu_clr
Idec pipeline
Br_clr
Br
Reg
file
24
ALU
22
Flags
Resull

DECLARATION FOR UTILITY OR DESIGN PATENT APPLICATION (37 CFR 1.63)

[ ] Declaration Submitted with Initial Filing OR [X] Declaration Submitted after Initial Filing (surcharge (37 CFR 1.16 (e)) required)

Attorney Docket Number	1406/52
First Named Inventor	Nie, Xiaoning
COMPLETE IF KNOWN	
Application Number	10/088,988
Filing Date	March 25, 2002
Art Unit	
Examiner Name	

As the below named inventor, I hereby declare that:

My residence, mailing address, and citizenship are as stated below next to my name.

I believe I am the original and first inventor of the subject matter which is claimed and for which a patent is sought on the invention entitled:

METHOD AND DEVICE FOR PROCESSING CONDITIONAL JUMP INSTRUCTIONS IN A PROCESSOR WITH PIPELINED ARCHITECTURE

(Title of the Invention)

the specification of which

[ ] is attached hereto

OR

[X] was filed on (MM/DD/YYYY) 03/25/2002 as United States Application Number or PCT International

Application Number **10/088,988** and was amended on (MM/DD/YYYY) ______ (if applicable).

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by any amendment specifically referred to above.

I acknowledge the duty to disclose information which is material to patentability as defined in 37 CFR 1 56, including for continuation-in-part applications, material information which became available between the filing date of the prior application and the national or PCT international filing date of the continuation-in-part application.

I hereby claim foreign priority benefits under 35 U.S.C. 119(a)-(d) or (f), or 365(b) of any foreign application(s) for patent, inventor's or plant breeder's rights certificate(s), or 365(a) of any PCT international application which designated at least one country other than the United States of America, listed below and have also identified below, by checking the box, any foreign application for patent, inventor's or plant breeder's rights certificate(s), or any PCT international application having a filing date before that of the application on which priority is claimed.

Prior Foreign Application Number(s)	Country	Foreign Filing Date (MM/DD/YYYY)	Priority Not Claimed	Certified Copy Attached? YES	NO
199 45 940.1	Germany	09/24/1999	[ ]	[ ]	[X]
PCT/EP00/09267	WIPO	09/21/2000	[ ]	[ ]	[X]
			[ ]	[ ]	[ ]
			[ ]	[ ]	[ ]

[ ] Additional foreign application numbers are listed on a supplemental priority data sheet PTO/SB/02B attached hereto:

PTO/SB/01 (10-01)
Approved for use through 10/31/2002 OMB 0651-0032
U S Patent and Trademark Office, U S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number

DECLARATION — Utility or Design Patent Application

Direct all correspondence to. [X] Customer Number or Bar Code Label 25297 PATENT TRADEMARK OFFICE		OR [ ] Correspondence address below	
Name			
Address			
City		State	ZIP
Country	Telephone		Fax

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under 18 U.S.C. 1001 and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

NAME OF SOLE OR FIRST INVENTOR : [ ] A petition has been filed for this unsigned inventor

Given Name (first and middle [if any]) 1-00 Xiaoning		Family Name or Surname Nie	
Inventor's Signature			Date May 6th, 2002
Residence: ~~City~~ München	State	Country DE	Citizenship DE DEX
Mailing Address Brehmstrasse 10			
City München	State	ZIP D-81543	Country Germany

NAME OF SECOND INVENTOR: [ ] A petition has been filed for this unsigned inventor

Given Name (first and middle [if any])		Family Name or Surname	
Inventor's Signature			Date
Residence: City	State	Country DE	Citizenship
Mailing Address			
City	State	ZIP	Country

[ ] Additional inventors are being named on the ____supplemental Additional Inventor(s) sheet(s) PTO/SB/02A attached hereto.